LDPC codes over rings for PSK modulation

Sridhara, D ; Fuja, T

Abstract: This paper describes the design and analysis of low-density parity-check (LDPC) codes over rings and shows how these codes, when mapped onto appropriate signal constellations, can be used to effect bandwidth-efficient modulation. Specifically, LDPC codes are constructed over the integer rings Zm and G2m and mapped onto phase-shift keying (PSK)-type signal sets to yield geometrically uniform signal space codes. This paper identifies and addresses the design issues that affect code performance. Examples of codes over Z8 and G64 mapped onto 8-ary and 64-ary signal sets at a spectral efficiency of 1.5 and 2.0 bits per second per hertz (b/s/Hz) illustrate the approach; simulation of these codes over the additive white Gaussian noise (AWGN) channel demonstrates that this approach is a good alternative to bandwidth-efficient techniques based on binary LDPC codes—e.g., bit-interleaved coded modulation.

DOI: https://doi.org/10.1109/TIT.2005.853330

LDPC Codes Over Rings for PSK Modulation

Deepak Sridhara
Institut für Mathematik
Universität Zürich
CH-8057, Zürich, Switzerland.
email: sridhara@math.unizh.ch

Thomas E. Fuja
Department of Electrical Engineering
University of Notre Dame
Notre Dame, IN 46556, U.S.A.
email: tfuja@nd.edu

Abstract

This paper describes the design and analysis of low-density parity check (LDPC) codes over rings and shows how these codes, when mapped onto appropriate signal constellations, can be used to effect bandwidth-efficient modulation. Specifically, LDPC codes are constructed over the integer rings $\mathbb{Z}_m$ and $\mathbb{G}_{m^2}$ and mapped onto PSK-type signal sets to yield geometrically uniform signal space codes. This paper identifies and addresses the design issues that affect code performance. Examples of codes over $\mathbb{Z}_8$ and $\mathbb{G}_{64}$ mapped onto 8-ary and 64-ary signal sets at a spectral efficiency of 1.5 and 2.0 bps/Hz illustrate the approach; simulation of these codes over the additive white Gaussian noise channel demonstrates that this approach is a good alternative to bandwidth-efficient techniques based on *binary* LDPC codes – e.g., bit-interleaved coded modulation.

Submitted August 2003 to IEEE Transactions on Information Theory. Revised February 2005.

Index Terms

Belief-propagation decoding, coded modulation, low density parity check codes.

Supported in part by NSF Grants CCF-02-05310 and EEC-02-03366 and the Indiana Twenty-First Century Research and Technology Fund. Some of this material was presented at the 2002 Information Theory Workshop in Bangalore, India.

I. INTRODUCTION

The "re-discovery" of low density parity check (LDPC) codes with near-capacity performance over binary channels has generated considerable activity in the coding community. The application of LDPC principles to denser signal sets has been examined in [1]-[2]. In those papers, binary LDPC codes are mapped to dense (non-binary) signal sets via multilevel and/or bit-interleaved coded modulation.

In this paper, a wholly different approach to designing LDPC-based coded modulation is examined; the proposed approach is to design non-binary LDPC codes that are appropriate for non-binary signal sets. Specifically, the LDPC codes are designed over groups that "match" the signal constellations.

Research in this area has its roots in Slepian's 1968 paper, "Group Codes for the Gaussian Channel" [3], which considered the design of signal sets generated by a group of orthogonal matrices acting on an "initial point" in the constellation. In [4], Forney formulated the essential rules for designing signal space codes that are geometrically well-structured and possessing the properties of Slepian-like codes. Around the same time, Loeliger [5] [6] introduced the concept of signal sets *matched* to groups. The codes constructed by Forney and Loeliger are uniform with respect to Euclidean distance; this means that the set of Euclidean distances between a reference codeword and all other possible codewords is independent of the choice of the reference codeword, and all the Voronoi (or maximum-likelihood decision) regions around each codeword are isomorphic. These codes are therefore appropriately referred to as geometrically-uniform (GU) codes.

More recently, Caire and Biglieri [7] addressed the design of block codes over the additive group $\mathbb{Z}_m$ and the signal space codes obtained when the resulting codewords are mapped onto phase shift keying (PSK) modulation signal sets. The focus of [7] was the structural properties of such codes, using a few short-blocklength codes over $\mathbb{Z}_4$ mapped onto 4-PSK as examples.

This paper uses the same basic approach as Caire and Biglieri – designing codes over cyclic groups and mapping them onto PSK signal sets – but it focuses on LDPC codes over those cyclic groups. The performance

of the resulting signal space codes are assessed when they are used to transmit information over the additive white Gaussian noise (AWGN) channel and decoding is carried out via a (suitably modified) version of message passing. Design issues that affect performance are addressed.

In related work, Davey and Mackay [8] designed LDPC codes over non-binary *fields* and applied these codes to binary modulated channels. One obvious way to effect bandwidth-efficient modulation would be to map the Davey/Mackay LDPC codes over GF(q) onto q-ary signal sets. However, GF(q) is not matched to any signal constellation for non-prime $q > 2$, and so the signal space codes thus constructed are not geometrically uniform. (For instance, if a code over GF(q) is mapped onto a q-ary constellation for $q > 2$, then the code's performance depends on the codewords that are transmitted.)

This paper is organized as follows. Section II reviews the concept of "matching" between a group and a signal set, and a few examples are given. In Section III, the structure of LDPC codes over the integer ring $\mathbb{Z}_m$ is addressed; in particular, the focus is on power-of-prime values of m and signal sets that are m-PSK or "PSK-like." Design issues that are addressed include threshold computation, selecting a degree distribution, and choosing the non-zero elements of the parity check matrix. Section IV uses the principles from Section III to design and assess ring-based LDPC codes mapped to dense signal constellations, and their performance is compared to systems that use binary LDPC codes and bit-interleaved modulation. In Section V, LDPC codes over Gaussian integer rings are constructed and their codewords are mapped onto appropriate 4- and 6-dimensional signal constellations. Section VI summarizes the results of this paper.

II. Modulation signal sets matched to groups

The following definition is due to Loeliger [5].

Definition 2.1: A signal set S is said to be *matched* to a group G if there exists a mapping $\mu : G \to S$ such that for all a and b in G, the following property holds:

$$d_E^2(\mu(a),\mu(b)) = d_E^2(\mu(b^{-1}\cdot a),\mu(e)) \tag{1}$$

where $d_E^2(x,y)$ is the squared Euclidean distance between x and y, e is the identity element of the group, and '$\cdot$' is the group operation.

By applying the mapping component-wise, Definition 2.1 may be extended to the direct product group G^n; thus, G^n is matched to S^n by the direct product mapping μ^n:

$$\mu^n(g_1,\ldots,g_n) \triangleq (\mu(g_1),\ldots,\mu(g_n)), \quad (g_1,\ldots,g_n) \in G^n$$

Let C be a subgroup of G^n, and let $\mu : G \to S$ be a matched mapping between G and S. Then the resulting signal space code $\mathbb{C} = \mu^n(C)$ is *geometrically uniform* [4]. It is easily verified that the squared Euclidean distance between any two arbitrary codewords in $\mathbb{C}$ – call them $\mathbb{C}_0 = \mu^n(C_0)$ and $\mathbb{C}_1 = \mu^n(C_1)$ – is given by

$$d_E^2(\mathbb{C}_0,\mathbb{C}_1) = d_E^2(\mathbb{C}_2,\mu^n([e,\ldots,e])),$$

where $\mathbb{C}_2 = \mu^n(C_1^{-1}\cdot C_0)$ and e is the group's identity element.

Now consider the m-PSK modulation signal constellation consisting of m points equally spaced around a circle in 2-D space; these m signal points may be represented as complex numbers on the unit-circle. Let $\mathbb{Z}_m$ denote the integers $\{0,1,\ldots,m-1\}$ under addition modulo m; then $\mathbb{Z}_m$ is isomorphic to the commutative group $\mathbb{Z}/m\mathbb{Z}$, and there exists a matched mapping μ from $\mathbb{Z}_m$ to the m-PSK modulation signal set given by

$$\mu(k) = \exp(j2\pi k/m), \qquad k \in \mathbb{Z}_m. \tag{2}$$

$\mathbb{Z}_m$ has a natural ring structure, so a linear block code over $\mathbb{Z}_m$ with code length n forms a $\mathbb{Z}$-submodule (and therefore a subgroup) of $(\mathbb{Z}_m)^n$. The signal space code obtained from a linear block code over $\mathbb{Z}_m$ with the matched mapping between $\mathbb{Z}_m$ and m-PSK is therefore geometrically uniform.

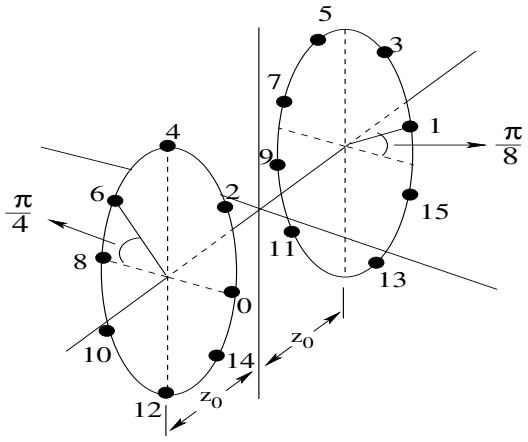

Fig. 1. $\mathbb{Z}_{16}$ matched to the three-dimensional constellation S_{16}.

There is another constellation matched to $\mathbb{Z}_m$ for *even* m – the three dimensional signal set S_m in Figure 1 [5]. This constellation is made up of two $(m/2)$-PSK constellations rotated by $2\pi/m$ radians relative to each other. The labeling in the figure indicates the matched mapping.

Finally, we will also construct some signal space codes over the 4-dimensional constellation $2\times$ (m-PSK) and the 6-dimensional constellation $2 \times S_m$, so we will need a matched mapping to those spaces from a ring with elements in $\mathbb{Z}_m \times \mathbb{Z}_m$. Of course, one possibility would be to simply use the ring of pairs from $\mathbb{Z}_m$ with operations defined component-wise; however, any linear code over this ring would be equivalent to a linear code over $\mathbb{Z}_m$ with the same block length. (Each codeword over $\mathbb{Z}_m \times \mathbb{Z}_m$ would be *two* codewords over $\mathbb{Z}_m$.) So instead we will consider the ring $\mathbb{G}_{m^2}$, with elements from $\mathbb{Z}_m \times \mathbb{Z}_m$ and operations defined as for complex numbers – i.e., $(a, b) + (c, d) = (a + c, b + d)$ and $(a, b) \cdot (c, d) = (ac - bd, ad + bc)$, where $a, b \in \mathbb{Z}_m$ and all operations are modulo m. This ring is isomorphic to the quotient ring $\mathbb{G}/(m + j0)\mathbb{G}$, where $\mathbb{G} = \{a + jb \mid a, b \in \mathbb{Z}\}$ is the ring of Gaussian integers. By constructing linear block codes over $\mathbb{G}_{m^2}$ and mapping the codewords componentwise onto $2\times$ (m-PSK) and $2 \times S_m$, the results are 4- and 6-dimensional geometrically uniform signal space codes.

III. LDPC CODES OVER $\mathbb{Z}_m$ MAPPED TO PSK SIGNAL SETS

This section addresses the construction of LDPC codes over the ring $\mathbb{Z}_m$. It begins by briefly reviewing the results from [7] and [9] that illuminate the structure of linear block codes over that ring; it then illustrates how those results can be applied specifically to the construction of LDPC codes. This section also considers the design issues that are important when the resulting LDPC codewords are mapped onto PSK modulation sets.

A. Structure of linear block codes over $\mathbb{Z}_m$

Because the ring $\mathbb{Z}_m$ does not possess all the structure of a finite field, one must be careful about how one characterizes a "linear code" over $\mathbb{Z}_m$. Fortunately, the following theorem by Caire and Biglieri [7] offers the necessary insight.

Theorem 3.1: Let $C \subset \mathbb{Z}_m^n$. Then the following statements are equivalent:

1) C is a subgroup of $\mathbb{Z}_m^n$
2) There exists an integer r $(0 \leq r \leq n)$, a set of linearly independent vectors $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_r\} \subset \mathbb{Z}_m^n$, and a set of nested ideals of $\mathbb{Z}_m$ (not necessarily distinct)

$$\mathbb{Z}_m \ > \ a_1\mathbb{Z}_m \ > \ a_2\mathbb{Z}_m \ > \ \ldots \ > \ a_r\mathbb{Z}_m \ > \ \{0\}$$

such that C can be written as the direct sum

$$C = \bigoplus_{i=1}^{r} a_i \mathbb{Z}_m \mathbf{x}_i$$

Moreover, the ideals and r are uniquely determined by C and m.

The proof of this theorem relies on the fact that $\mathbb{Z}$ is a principal ideal domain; hence, the code C forms a $\mathbb{Z}$-submodule of the free $\mathbb{Z}$-module $\mathbb{Z}_m^n$. The invariant factor theorem for modules states that for a principal ideal domain R, any submodule of a free R-module can be decomposed uniquely into a direct sum of R-modules as described in Theorem 3.1.

Theorem 3.1 provides the mechanism required to construct a generator matrix for a linear block code over $\mathbb{Z}_m$. Focusing on the case $m = p^a$ for a prime p and positive integer a (see also [9]), then a non-trivial linear code C of block length n over $\mathbb{Z}_{p^a}$ can be expressed as

$$C = \{[v_0 \ v_1 \ v_2 \ \ldots \ v_{a-1}]\, G : v_i \in \mathbb{Z}_{p^{a-i}}^{k_i}\}$$

where the generator matrix G is in the form

$$G = \begin{bmatrix} I_{k_0} & A_{0,1} & A_{0,2} & A_{0,3} & \ldots & A_{0,a-1} & A_{0,a} \\ 0 & pI_{k_1} & pA_{1,2} & pA_{1,3} & \ldots & pA_{1,a-1} & pA_{1,a} \\ 0 & 0 & p^2 I_{k_2} & p^2 A_{2,3} & \ldots & p^2 A_{2,a-1} & p^2 A_{2,a} \\ . & . & . & . & \ldots & . & . \\ 0 & 0 & 0 & 0 & \ldots & p^{a-1} I_{k_{a-1}} & p^{a-1} A_{a-1,a} \end{bmatrix}. \tag{3}$$

Here, the k_i's are non-negative integers such that $k_0 + k_1 + \ldots + k_a = n$, and $A_{i,j}$ is a $k_i \times k_j$ matrix with elements from $\mathbb{Z}_{p^a}$. Also, I_j is the $j \times j$ identity matrix.

This linear block code C over $\mathbb{Z}_{p^a}$ contains p^k codewords, where $k = \sum_{i=0}^{a-1}(a-i)k_i$, and so the rate of the code is

$$R = \frac{k}{a\ n} = \frac{1}{n}(k_0 + \frac{a-1}{a}k_1 + \frac{a-2}{a}k_2 + \ldots + \frac{1}{a}k_{a-1}).$$

This observation demonstrates the existence of an "almost systematic" generator matrix for any linear code over $\mathbb{Z}_{p^a}$ – unlike the case for linear codes over a finite field, where a systematic generator is always guaranteed.

Definition 3.1: A code C over $\mathbb{Z}_{p^a}$ is a free $\mathbb{Z}_{p^a}$ module (and is said to be *free*) if $k_1 = k_2 = \ldots = k_{a-1} = 0$.

So a free code over $\mathbb{Z}_{p^a}$ has a systematic generator matrix.

The defining characteristic of a low density parity check code is, of course, the parity check matrix. If

the dual of a code C over $\mathbb{Z}p^a$ is defined to be the set $C^\perp = \{\mathbf{x} \in \mathbb{Z}_{p^a}^n \ : \ \mathbf{x} \cdot \mathbf{y} = 0, \forall \mathbf{y} \in C\}$, where $\mathbf{x} \cdot \mathbf{y} \triangleq x_1 y_1 + \ldots + x_n y_n$, then [9] showed that $C^\perp$ has a generator matrix of the form

$$G^\perp = \begin{bmatrix} B_{0,a} & B_{0,a-1} & \ldots & B_{0,3} & B_{0,2} & B_{0,1} & I_{k_a} \\ pB_{1,a} & pB_{1,a-1} & \ldots & pB_{13} & pB_{1,2} & pI_{k_{a-1}} & 0 \\ p^2B_{2,a} & p^2B_{2,a-1} & \ldots & p^2B_{2,3} & p^2I_{k_{a-2}} & 0 & 0 \\ . & . & \ldots & . & . & . & . \\ p^{a-1}B_{a-1,a} & p^{a-1}I_{k_1} & \ldots & 0 & 0 & 0 & 0 \end{bmatrix}. \tag{4}$$

The k_i's in expressions (3) and (4) are the same, and the $B_{i,j}$ matrices in (4) can be obtained from the $A_{i,j}$ matrices in (3). The space spanned by the rows of $G^\perp$ is the *dual* of the space spanned by the rows of G, and so $G^\perp$ is a parity check matrix for C. Elementary row and column operations on $G^\perp$ yield additional parity check matrices.

The structural properties described above indicate how one can begin with a parity check matrix H for a code over $\mathbb{Z}_m$ and construct a generator matrix for the same code. An example follows.

Example: Consider a code C over $\mathbb{Z}_8$ (so $p = 2$ and $a = 3$) with parity check matrix

$$H = \begin{bmatrix} 7 & 5 & 3 & 1 & 0 \\ 4 & 4 & 1 & 0 & 1 \\ 0 & 6 & 4 & 0 & 2 \end{bmatrix}$$

If we replace the third row with the sum of the third row and six times the second row we obtain another parity check matrix for the same code

$$H_1 = \begin{bmatrix} 7 & 5 & 3 & 1 & 0 \\ 4 & 4 & 1 & 0 & 1 \\ 0 & 6 & 2 & 0 & 0 \end{bmatrix} = \left[\begin{array}{cc|c|cc} 7 & 5 & 3 & 1 & 0 \\ 4 & 4 & 1 & 0 & 1 \\ \hline 2(0) & 2(3) & 2(1) & 0 & 0 \end{array}\right]$$

Observe that H_1 is in the form of equation (4):

$$H_1 = \begin{bmatrix} B_{03} & B_{02} & B_{01} & I_{k_3} \\ 2B_{13} & 2B_{12} & 2I_{k_2} & 0 \\ 4B_{23} & 4I_{k_1} & 0 & 0 \end{bmatrix}$$

where $k_3 = 2$, $k_2 = 1$, $k_1 = 0$, $n = 5$, and $k_0 = n - k_1 - k_2 - k_3 = 2$. Therefore, B_{01} is a $k_3 \times k_2 = 2 \times 1$ matrix, B_{03} is a $k_3 \times k_0 = 2 \times 2$ matrix, and B_{13} is a $k_2 \times k_0 = 1 \times 2$ matrix. B_{02}, B_{12}, and B_{23} are all void because $k_1 = 0$.

Therefore, we obtain the following generator matrix for this code:

$$G = \begin{bmatrix} I_{k_0} & A_{01} & A_{02} & A_{03} \\ 0 & 2I_{k_1} & 2A_{12} & 2A_{13} \\ 0 & 0 & 4I_{k_2} & 4A_{23} \end{bmatrix} = \left[\begin{array}{cc|c|cc} 1 & 0 & 0 & 1 & 4 \\ 0 & 1 & 5 & 4 & 7 \\ \hline 0 & 0 & 4(1) & 4(1) & 4(1) \end{array}\right]$$

where, $A_{23} = -B_{01}^T$, $A_{01} = A_{12} = A_{13} = 0$ (since $k_1 = 0$), $A_{02} = -B_{13}^T$, and $A_{03} = B_{13}^T B_{01}^T - B_{03}^T$.

So the cardinality of C is $|C| = (8)^{k_0}(8/2)^{k_1}(8/4)^{k_2} = 128$ and the rate is $R = \log_8(128)/5 = 7/15$.

B. The effect of zero-divisors in H on minimum distance

An obvious difference between designing a binary LDPC code and designing a non-binary LDPC code is in the selection of the non-zero entries of the parity check matrix. Intuitively, the presence of zero-divisors[1] in H are problematic; for instance, if all the non-zero entries in a column are zero-divisors, then the resulting code will have a minimum distance of one!

The following theorem describes a relation between the minimum Hamming distance of a code over $\mathbb{Z}_{p^a}$ and that of a related code over the field $\mathbb{Z}_p$. This theorem yields some insight into the role of zero-divisors in H.

[1]Recall that a zero-divisor is a non-zero ring element a such that $a \cdot b = 0$ for some non-zero element b. The zero-divisors in $\mathbb{Z}_{p^a}$ are the multiples of p.

Theorem 3.2: Let H be a parity check matrix of a code C over the ring $\mathbb{Z}_{p^a}$. Let H_p be the parity check matrix of a code C_p over the field $\mathbb{Z}_p$ obtained by reducing every entry in H modulo p. Then the minimum Hamming distance of the two codes are related by $d_H(C) \leq d_H(C_p)$, with equality if C is a free $\mathbb{Z}_{p^a}$ module, as defined in Definition 3.1.

The proof of Theorem 3.2 is given in the appendix.

Of course, the zero-divisors in H are exactly the non-zero elements of H that are reduced to zero in H_p. Therefore, placing zero-divisors in H will decrease the number of non-zero elements in H_p, and to the extent that results in a low-minimum distance code C_p, the code C is likewise compromised.

For example: Suppose H describes a $(3, 9)$ regular LDPC code over $\mathbb{Z}_8$. Then if each column of H contains one zero-divisor – one even number – among its three non-zero entries and each row of H contains three zero-divisors among its nine non-zero entries, then the reduced matrix H_2 is the parity check matrix for a $(2, 6)$ binary LDPC code. It is well known that the minimum distance of binary $(2, k)$ LDPC codes grows at most logarithmically with block length [10], and Theorem 3.2 indicates that the minimum distance of the $(3, 9)$ LDPC code over $\mathbb{Z}_8$ would exhibit similarly anemic growth if the zero-divisors were distributed as described.

C. BP decoding and thresholds of LDPC codes over $\mathbb{Z}_{p^a}$

The message passing (or belief propagation (BP)) decoder as originally formulated for binary LDPC codes can be readily modified to accommodate non-binary alphabets [8]. Each iteration of the BP decoder consists of two half-cycles

- the computation of messages produced by the constraint nodes based on the messages generated by the variable nodes; we shall refer to these as "constraint-output" messages.
- the computation of messages produced by the variable nodes based on the messages generated by the constraint nodes; we shall refer to these as the "constraint-input" messages.

For binary codes each message is a real number representing a (log-)likelihood ratio. For codes over m-ary alphabets the messages are vectors representing a probability distribution over $\{0, 1, \ldots, m-1\}$. The initialization and update rules for message passing applied to codes over $\mathbb{Z}_m$ are similar to those for codes over the finite field GF(m) as described by Mackay and Davey in [8]. A detailed description of message passing over $\mathbb{Z}_m$ and a discussion of its complexity for the case $m = 8$ is given in [11]; in particular, it is shown that the computational complexity of message passing applied to a blocklength-n code over $\mathbb{Z}_8$ is comparable to that of message passing applied to a blocklength-$3n$ binary code.

The BP decoder converges when the constraint-output messages converge to a deterministic probability distribution. It has been shown [12] that this happens when the channel signal to noise ratio (SNR) is above a *threshold* value, assuming the constraint graph is cycle-free. This section describes a method for estimating the threshold of a BP decoder operating on a cycle-free graph for an LDPC code over a ring when the channel is AWGN.

To estimate the threshold, assume that the all-zero codeword is modulated and transmitted – i.e., the signal point $\mu(0)$ is repeatedly sent. A large number (thousands) of constraint nodes are simulated during each iteration; the inputs to these constraint nodes are initially determined by the channel observations, and the messages that they produce are used to construct an estimate of the probability distribution of the resulting constraint-output message – i.e., a histogram. This histogram is then used to generate i.i.d. realizations of the constraint-output messages, and these i.i.d. realizations are used to compute a set of constraint-input messages via the "variable node calculation". Once the constraint-input messages are computed, the constraint node calculations are carried out and a new round of constraint-output messages are generated, producing another histogram. The process continues until convergence takes place or the maximum number of iterations is reached.

For binary codes and the BPSK-modulated AWGN channel, the constraint-output message distribution can be approximated by a *consistent* Gaussian distribution, i.e., one with a variance that is half its mean. Thus,

tracking the message distribution reduces to tracking the mean of a Gaussian random variable [13].

With non-binary LDPC codes things are not quite so simple. A constraint-output message $\mathbf{r}$ is an m-tuple $[r_0, r_1, \ldots, r_{m-1}]$, where r_i is the probability of the constraint being satisfied when the associated variable node assumes a value of i. We will use the entropy of $\mathbf{r}$, given by $h(\mathbf{r}) = -\sum_{i=0}^{m-1} r_i \log_2(r_i)$ to measure the quality of $\mathbf{r}$ – i.e., the smaller the entropy of $\mathbf{r}$, the closer $\mathbf{r}$ is to convergence.

So calculating the threshold requires repeated i.i.d. realizations of the constraint-output message $\mathbf{r}$, where the probability distribution on the random vector $\mathbf{r}$ during the i^{th} iteration is estimated empirically from the outputs of the variable node calculations performed during the i^{th} iteration; and those variable node calculations use as inputs the i.i.d. realizations from the previously estimated probability distribution on $\mathbf{r}$, i.e., from the $(i-1)^{st}$ iteration.

However, for $m > 2$ estimating the probability distribution on the constraint-output becomes infeasible; therefore, rather than generating a histogram to describe the multidimensional distribution of $\mathbf{r}$ for each iteration, the inputs to the variable node calculations are sampled from the constraint-output message values computed during the previous iteration. For example, if 10^4 constraint nodes in a (j, k) regular graph are simulated in each iteration, they will produce $10^4 \cdot k$ constraint-output messages; in the next half-iteration, for each variable node calculation we sample from these $10^4 \cdot k$ messages $j - 1$ of them to serve as the inputs.

For the threshold estimates reported in this paper, fifty (50) such iterations were completed, and if the average entropy of the constraint-output messages fell below 0.001 bits then the algorithm was deemed to have converged. The smallest signal to noise ratio for which convergence took place is referred to as the threshold of the constraint graph.

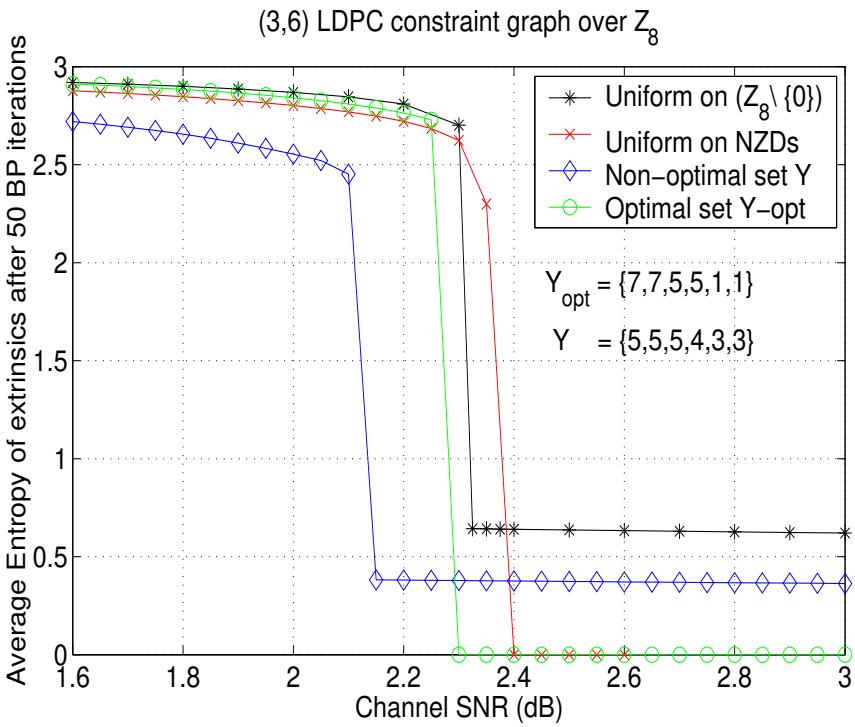

Fig. 2. Average entropy of the constraint-output messages for different weight sets for $(3, 6)$ LDPC code mapped to 8-PSK.

D. Convergence and the choice of non-zero entries of H

In Section III-B it was shown that the choice of the non-zero entries in H – and in particular the use of zero-divisors in H – has an effect on the minimum distance of the resulting code. In this section we consider the effect of the choice of non-zero entries in H on the convergence of the BP decoder.

Consider first a regular (j, k) LDPC code over $\mathbb{Z}_m$. All the constraint nodes have k incident edges, and we begin by assuming that all the constraint nodes have the same set of k weights associated with those edges; this is equivalent to assuming that each row of H contains the same k non-zero entries. There are $\binom{m-2+k}{k}$ possible choices for the k non-zero entries, and for each such set $Y \in (\mathbb{Z}_m \backslash \{0\})^k$ one can carry out the threshold computation and thereby find an optimal set Y_{opt} – i.e., the edge weights that result in convergence at the lowest channel SNR. Recall that "convergence" means that the entropy of the constraint-output messages tends to zero. (The optimal set(s) of weights was found from an exhaustive search over $\binom{m+2-k}{k}$ possible choices.)

Figure 2 shows the average entropy of the constraint-output messages after 50 BP iterations as a function of the channel SNR. The LDPC code in this simulation is a $(3, 6)$ regular LDPC code over $\mathbb{Z}_8$, with the codewords mapped to the 8-PSK constellation prior to transmission over an AWGN channel. There were several "optimal" weight sets, and they all contained *only* non-zero-divisors; each code based on an optimal weight set converged at $\mathrm{SNR} = 2.3$ dB. One of those optimal weight sets was $\{7, 7, 5, 5, 1, 1\}$, and the average entropy for this code is shown in the figure. Also included in Figure 2 is the behavior observed under the following scenarios:

- all the non-zero entries are picked i.i.d. equiprobably from the non-zero-divisors $\{1, 3, 5, 7\}$; this results in a slightly higher threshold of 2.4 dB.
- all the non-zero entries are picked i.i.d. equiprobably from the non-zero entries of the ring $\mathbb{Z}_m$ – i.e., from $\{1, 2, 3, 4, 5, 6, 7\}$. In this case there was no convergence over the range of SNR values simulated. However, we do notice a precipitous drop in average entropy at $\mathrm{SNR} = 2.3$ dB – not down to zero but to approximately 0.6 bits.
- finally, the figure also includes the average entropy that results when the weight sets are all $\{5, 5, 5, 4, 3, 3\}$. Note that this set, which includes the zero-divisor 4, once again results in a code that does not converge; however, the average entropy *does* (once again) drop precipitously to a low (albeit non-zero) value. Moreover, this drop occurs at a SNR (2.15 dB) where the "optimal" code has an average entropy that is still quite high. This suggests that zero-divisors can help the decoder to *begin* to converge, but they may inhibit completion.

The above observations suggest using a mixture of weight sets, Y_{opt} and Y; the constraint nodes with weight set Y can initiate convergence quicker, while the nodes with weight set Y_{opt} may help complete the convergence of the decoder. This principle is akin to the "doping" idea in [14] and is discussed briefly in the first bullet at the end of Section IV-C.

IV. Coded modulation based on LDPC codes over $\mathbb{Z}_8$

Using the principles developed in Section III, we now design LDPC codes over $\mathbb{Z}_8$ and map the resulting codewords onto the 8-PSK signal set. We begin by designing *regular* LDPC codes and restrict the entries in H to be non-zero-divisors. Then, the constraint graphs are made irregular with an edge profile based on the threshold. We then observe how the choice of the edge-weights affects the performance of the BP decoder. The codes' performance on the AWGN channel is compared with that of bit-interleaved coded modulation incorporating binary LDPC codes.

A. Regular LDPC constraint graphs over $\mathbb{Z}_8$

We begin with a very simple design: Construct a binary regular $(3, 9)$ LDPC matrix H, and replace the ones in H with the non-zero-divisors of $\mathbb{Z}_8$ (i.e., 1,3,5,7) chosen i.i.d. equiprobably.

Three LDPC codes of lengths 3,000, 10,000, and 50,000 were thus designed. Generator matrices for the three codes were obtained via the procedure in Section III, and rate 2.0 bps/Hz geometrically uniform signal space codes were obtained by mapping the resulting codewords onto the 8-PSK signal set using the mapping of equation (2). The performance of the resulting codes is shown in Figure 3. Also included is the performance of bit-interleaved coded modulation (BICM) using binary LDPC codes[2] with Gray mapping; the decoding in this case is tandem decoding where demodulation and binary BP decoding are done separately. (That is, the demodulator first converts the channel observations into log-likelihood values for the bits of the binary LDPC codewords and BP decoding is then applied.) All the curves shown in the figure correspond to regular $(3, 9)$ LDPC codes. In this case, the non-binary group based codes are found to consistently outperform bit-interleaved coded modulation codes that use binary LDPCs; a gain of 0.10-0.15 dB is observed.

[2]The binary LDPC codes used for comparison were taken from the web site maintained by R. Neal at the University of Toronto, http://www.cs.toronto.edu/~radford/software-online.html

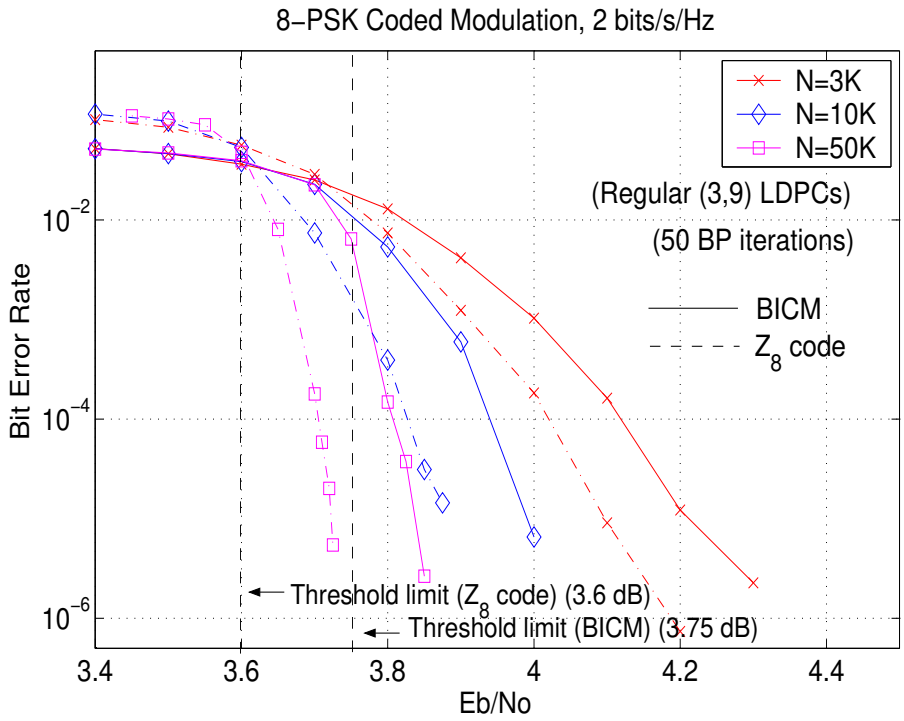

Fig. 3. Regular ring-based vs bit-interleaved LDPC with 8-PS modulation. (Rate = 2.0 bits/symbol.)

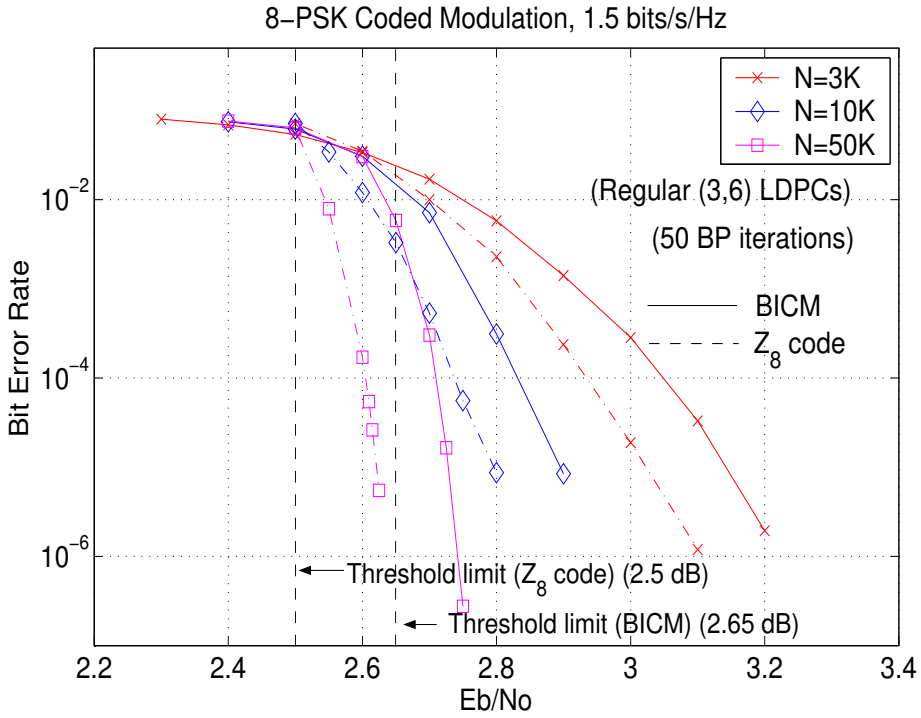

Fig. 4. Regular ring-based vs bit-interleaved LDPC with 8-PSK modulation. (Rate = 1.5 bits/symbol.)

The threshold of a $(3, 9)$ LDPC code over $\mathbb{Z}_8$ with non-zero elements of H selected i.i.d. from $\{1, 3, 5, 7\}$ and codewords mapped to 8-PSK is 3.6 dB, whereas the threshold of the BICM system using binary $(3, 9)$ codes Gray-mapped to 8-PSK is 3.75 dB. The performance of the block length 50,000 $(3, 9)$ LDPC code over $\mathbb{Z}_8$ in Figure 3 is about 0.12 dB from the threshold at a BER of 10^{-5}.

Figure 4 shows the analogous performance of regular $(3, 6)$ LDPC codes mapped to 8-PSK to effect 1.5 bps/Hz. Here again, the non-binary LDPC codes outperform the BICM scheme using binary LDPC codes by about 0.1 dB.

B. Irregular LDPC constraint graphs over $\mathbb{Z}_8$

The threshold calculation described in Section III-C can be used to optimize the degree profile of an LDPC code over $\mathbb{Z}_8$. Specifically, we determine the node degree distribution (λ, ρ) with the best threshold among the choices examined.

Recall that a code's degree profile is specified by two polynomials: $\lambda(x) = \sum_i \lambda_i x^{i-1}$ and $\rho(x) = \sum_i \rho_i x^{i-1}$, where λ_i and ρ_i are the fraction of edges incident on degree i variable nodes and constraint nodes, respectively. Since the thresholds are calculated using Monte-Carlo techniques, the search complexity is quite high; as a result, we looked over a relatively narrow range of degree profiles. Specifically: We fixed $\rho(x)$ to be a "concentrated" profile – i.e., of the form $\rho(x) = \rho_d x^{d-1} + \rho_{d+1} x^d$ – much as was done in [15]. Then, for a given $\rho(x)$ we searched for the $\lambda(x)$ yielding the best threshold over the range $\deg[\lambda(x)] \leq 11$ – i.e., we allowed the code bits to participate in at most twelve parity checks. Finally, for a given degree profile, a binary parity check matrix was randomly generated and then the non-zero elements of the binary matrix were replaced by the non-zero-divisors of $\mathbb{Z}_8$, chosen i.i.d. and equiprobably.

Using this search technique, the following profile was found to yield the best threshold among LDPC codes over $\mathbb{Z}_8$ with a rate of 2.0 bps/Hz when matched to 8-PSK modulation:

$$\lambda(x) = 0.43x + 0.2x^2 + 0.25x^3 + 0.12x^{11} \quad \text{and} \quad \rho(x) = 0.5x^7 + 0.5x^8. \tag{5}$$

The threshold for this degree distribution was $E_b/N_0 = 3.15$ dB.

Figure 5 shows the performance of irregular and regular LDPC codes over $\mathbb{Z}_8$ with matched 8-PSK modulation at three different block lengths. The codes marked "Irrg. 1" have the degree profile in (5). The irregular codes perform about 0.4 dB better than the regular codes in the waterfall region. However, the error floor is more pronounced for the irregular codes than for the regular codes.

Similarly, for a bandwidth efficiency of 1.5 bps/Hz, the following irregular degree-profile was found to yield the best threshold among the choices of LDPC codes over $\mathbb{Z}_8$ considered:

$$\lambda(x) = 0.327857x + 0.35x^2 + 0.05x^7 + 0.272143x^{11} \quad \text{and} \quad \rho(x) = 0.5x^5 + 0.5x^6, \tag{6}$$

The threshold for this degree distribution was $E_b/N_0 = 1.55$ dB.

Figure 6 shows the performance of 1.5 bps/Hz regular and irregular LDPC codes over $\mathbb{Z}_8$ at three different block lengths. The codes marked "Irrg. 2" have the degree profile in (6). There is a dramatic improvement (about 0.8 dB) in the performance of the irregular LDPCs over that of the regular $(3, 6)$ LDPC codes. The block length 50,000 Irrg. 2 code achieves a bit error rate of 10^{-6} at ~ 0.3 dB from the threshold limit, i.e., ~ 0.6 dB from capacity (1.25 dB).

C. Choosing the non-zero entries of the LDPC matrix H

In Section IV-A the non-zero entries of the LDPC matrices were chosen equiprobably from among the non-zero-divisors of $\mathbb{Z}_8$. However, we will now see that selecting these entries in ways motivated by the threshold calculation can improve the resulting code performance – at least for regular LDPC codes.

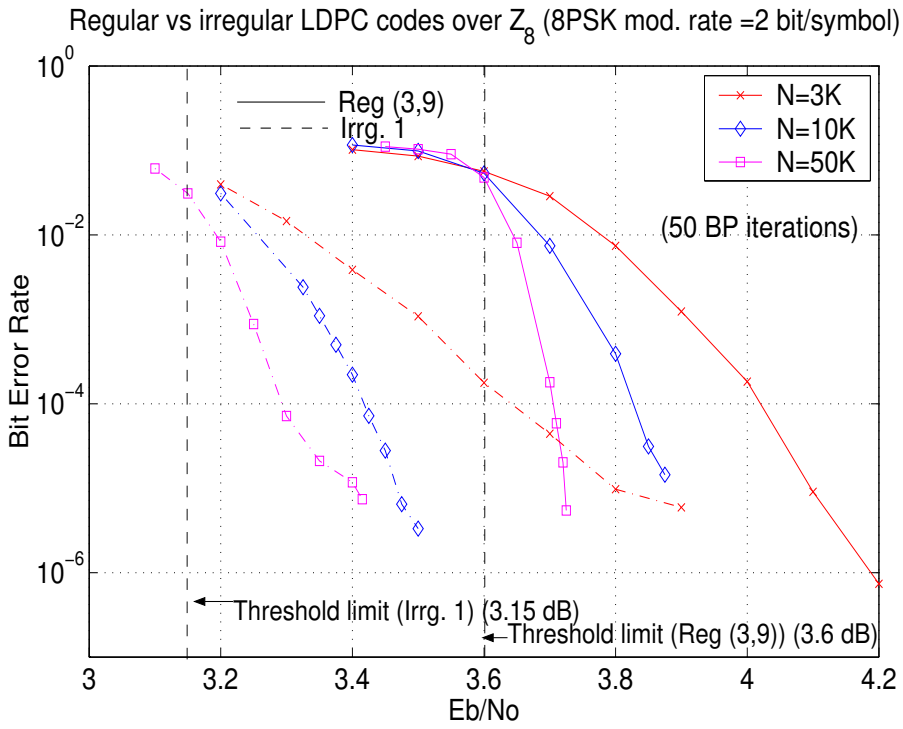

Fig. 5. Regular vs irregular LDPCs over $\mathbb{Z}_8$ with 8-PSK modulation. (Rate = 2 bits/symbol.)

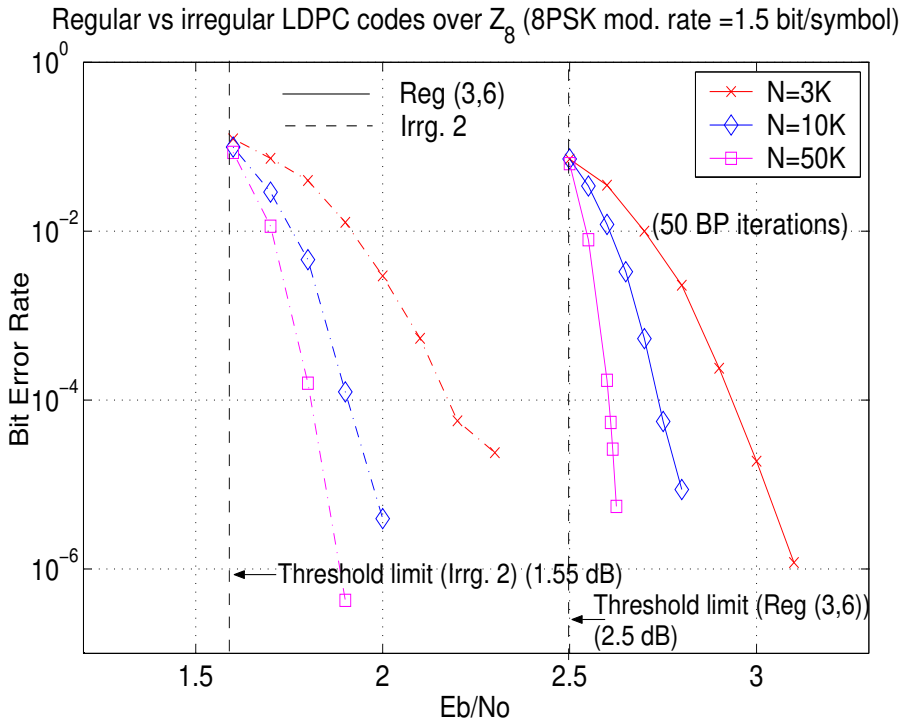

Fig. 6. Regular vs irregular LDPCs over $\mathbb{Z}_8$ with 8-PSK modulation. (Rate=1.5 bits/symbol.)

Looking back at Figure 2, we observe that choosing the weights on the edges of a regular $(3, 6)$ LDPC graph in four different ways leads to four different convergence behaviors. This motivates the construction of LDPC codes over $\mathbb{Z}_8$ based on a parity check matrix with non-zero elements selected according to four different strategies:

- Strategy S1: Each "1" in the binary parity check matrix is replaced with a non-zero element of $\mathbb{Z}_8$, selected i.i.d. and equiprobably.
- Strategy U1: Each "1" in the binary parity check matrix is replaced with a non-zero-divisor, selected i.i.d. and equiprobably.
- Strategy O1: the six ones in each row of the binary parity check matrix are replaced by the six elements of the set $Y_{\rm opt} = \{1, 1, 5, 5, 7, 7\}$ in random order.
- Strategy N1: the six ones in each row of the binary parity check matrix are replaced by the six elements of the set $Y = \{3, 3, 4, 5, 5, 5\}$ in random order.

Figure 7 shows the performance of 1.5 bps/Hz signal space codes over 8-PSK based on $(3, 6)$ LDPC codes over $\mathbb{Z}_8$ with the non-zero elements picked according to these four strategies. Two different block lengths are simulated – $N = 3,000$ and $N = 50,000$. It is observed that (as expected from the threshold calculations) strategy O1 outperforms strategy U1. Somewhat surprisingly, strategy N1 performs substantially better than either U1 or O1 until the decoder reaches an error floor.

Figure 8 shows similar results for 2.0 bps/Hz signal space codes over 8-PSK based on $(3, 9)$ LDPC codes over $\mathbb{Z}_8$. In this case, the "optimal" set of edge weights for each constraint node was $Y_{opt} = \{1, 1, 1, 1, 1, 3, 3, 7, 7\}$ and the set of edge weights that yielded non-convergent (but low-entropy) behavior was $Y = \{3, 3, 3, 3, 4, 4, 5, 5, 5\}$. The threshold for the constraint graph with edge weights Y_{opt} (i.e., adopting strategy O1) was 3.5 dB while the threshold for a graph with edge weights picked i.i.d. equiprobably from the non-zero-divisors (i.e., strategy U1) was 3.6 dB. The "dropoff" in average message entropy for the graph with edge weights Y (i.e., strategy

N1) occurred at 3.4 dB.

In Figures 7 and 8 significant coding gain is obtained when the non-zero elements are selected according to the non-convergent-but-low-entropy strategy "N1." It could be conjectured that this strategy might raise the error floor, and indeed some evidence of this is seen, at least in the 1.5 bps/Hz N=50,000 code and the 2.0 bps/Hz N=3,000 code. In these two cases the error floor is raised to a BER of approximately 10^{-6}. Another dramatic example of error floor is seen in the 1.5 bps/Hz N=50,000 case when the non-zero elements of H are picked i.i.d. from the non-zero elements of $\mathbb{Z}_8$

Before concluding this section, we make two brief observations on attempts to modify the LDPC code design in view of the results obtained so far.

- Strategies "O1" and "N1" replaced the non-zero elements in *each* row of the parity check matrix with the *same* set of k weights. It is natural to wonder if this restriction limits performance – if using different sets of k weights at different constraint nodes could improve performance. Simulations were carried out to test this conjecture. It was found that, when a "mixture" of strategies is employed, the resulting performance is similarly mixed; if a fraction p of the constraint nodes had edge weights selected according to one strategy and the rest were selected according to another strategy, the performance would lie in between the performances obtained by those two strategies applied solely. This approach could be used to trade off the best properties of two different strategies - e.g., the good "waterfall" of N1 versus the low error floor of O1.
- Finally, applying different strategies for selecting the non-zero elements of H had negligible effect on irregular codes with degree profiles already selected to minimize the threshold. It appears that almost all the performance gain possible is obtained by irregularizing the constraint graph, and little additional improvement can be expected by selecting the non-zero elements of the parity check matrix differently.

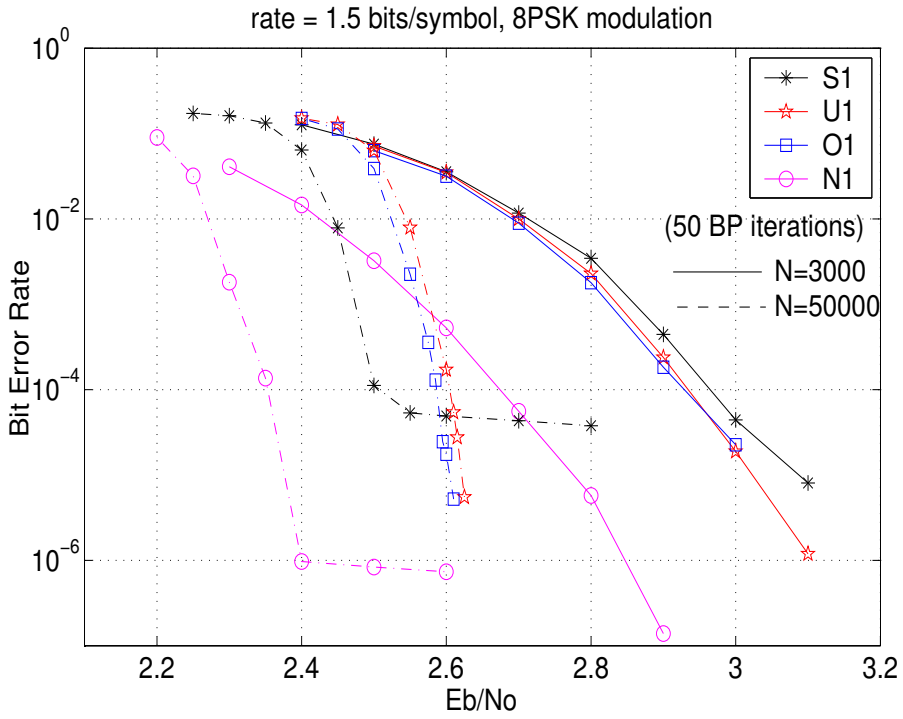

Fig. 7. Different strategies for choosing non-zero entries in a $(3, 6)$ LDPC matrix.

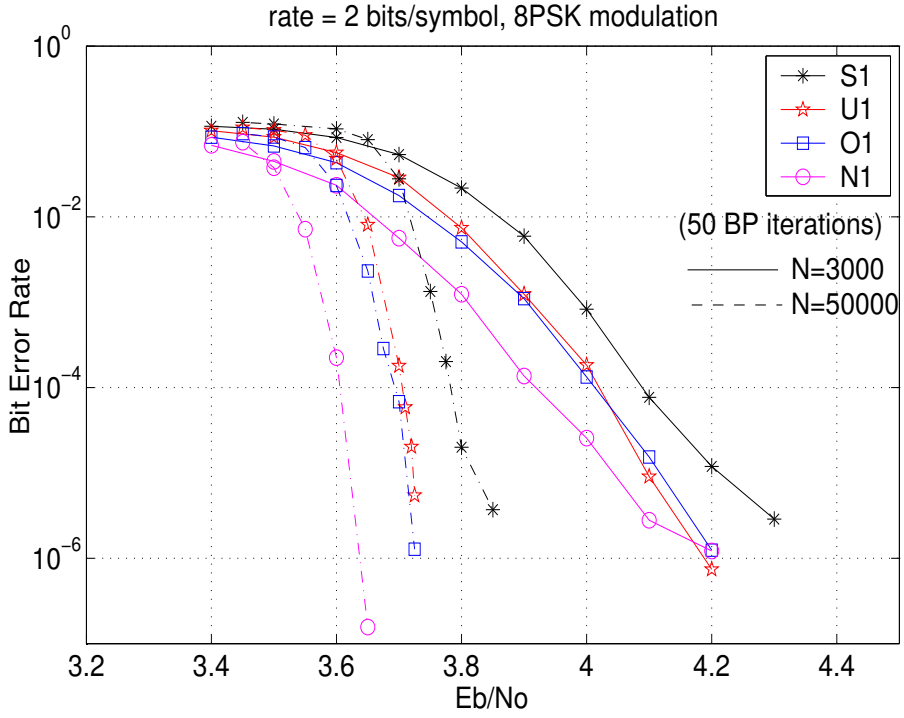

Fig. 8. Different strategies for choosing non-zero entries in a $(3, 9)$ LDPC matrix.

V. Coded modulation based on LDPC codes over the Gaussian integers

Section III-A describes how an encoder for a code over $\mathbb{Z}_m$ can be constructed from a (low-density) parity check matrix for the same. This technique can be extended to codes over other rings derived from principal ideal domains. In this section, we illustrate this by constructing some codes over the ring $\mathbb{G}_{64}$ derived from Gaussian integers. (Recall from Section II that the elements of $\mathbb{G}_{64}$ are two-tuples over $\mathbb{Z}_8$ (i.e., $|\mathbb{G}_{64}| = 64$) with addition performed component-wise modulo 8 and multiplication performed as for complex numbers modulo 8.)

In the examples in this section, 4500 information bits are represented with non-binary codewords and then modulated using a matched mapping.

In the first example, a binary regular $(3, 12)$ LDPC matrix of length 1000 is randomly constructed, and the non-zero entries of the matrix are replaced by the non-zero-divisors of $\mathbb{G}_{64}$, chosen i.i.d. and equiprobably. A generator matrix for the resulting LDPC code over $\mathbb{G}_{64}$ is obtained, and each code symbol is mapped onto the six-dimensional $2 \times S_8$ signal constellation, resulting in a rate of 4500 bits per 1000 6D-symbols or 1.5 bits/2D-symbol and an effective block length of 3000 2D-symbols. In a similar manner, a regular $(3, 6)$ LDPC matrix of length 1500 is constructed over $\mathbb{G}_{64}$. Its 2^{4500} codewords are mapped onto the four-dimensional 2×8-PSK signal constellation to effect a 1.5 bps/2D-symbol code, also with a block length of 3000 2D-symbols.

These two $\mathbb{G}_{64}$-LDPC codes may be decoded using the BP decoder. The complexity of BP decoding over $\mathbb{G}_{64}$ is significantly higher than that of binary LDPC codes. In contrast, BP decoding over $\mathbb{Z}_8$ can be performed quite efficiently using Fourier transform techniques.

The performance of these two $\mathbb{G}_{64}$-LDPC codes can be compared with that of LDPC codes over $\mathbb{Z}_8$ with comparable rates and block lengths. The ring structures of $\mathbb{Z}_8$ and $\mathbb{G}_{64}$ yield codes that perform quite differently with the same modulation. Figure 9 compares the performance of the two $\mathbb{G}_{64}$-codes with the performance of: (1) a regular $(3, 6)$ $\mathbb{Z}_8$-LDPC code mapped to 8-PSK, and (2) a regular $(3, 12)$ $\mathbb{Z}_8$-LDPC code mapped to the 3D S_o signal set. All four codes have a rate of 1.5 bits/2D-symbol and effective block lengths of 3000 2D

symbols; in each case, the non-zero elements of the parity check matrix are chosen i.i.d. and equiprobably from the appropriate set of non-zero-divisors.

Figure 9 indicates that the LDPC code designed over $\mathbb{G}_{64}$ with $2 \times S_8$ modulation performs quite well when compared with an analogous code designed over $\mathbb{Z}_8$. Surprisingly, however, the LDPC code over $\mathbb{G}_{64}$ with 2×8-PSK modulation performs poorly in comparison with an analogous code designed over $\mathbb{Z}_8$ at all SNRs.

Comparing the performance in terms of the modulation, it is observed that the codes mapped onto S_8 perform better than the codes mapped onto 8-PSK for low SNRs. This is because S_8 has a larger minimum squared Euclidean distance between symbol points than an equal energy 8-PSK; hence, better initial estimates are obtained for each symbol of a codeword mapped onto S_8, thereby improving the performance at low SNRs. However, recall that $(3, 12)$ LDPC codes were used with S_8, whereas $(3, 6)$ LDPC codes were used with 8-PSK. Since the constraint nodes in a $(3, 6)$ graph are better decoders than constraint nodes in a $(3, 12)$ graph, we observe in Figure 9 that the bit error rate of $(3, 12)$ LDPC codes (with S_8) falls off less sharply than that of $(3, 6)$ LDPC codes (with 8-PSK).

The figure also shows the performance of a code designed over the Galois ring $GR(2^3, 2)$. A Galois ring is a finite commutative ring with a unique maximal ideal; it is a generalization of a Galois field, and the Galois ring denoted $GR(p^n, r)$ contains p^{nr} elements. (See [16] for details.) Theorem 3.1 can be applied to Galois rings, making it possible to construct and decode LDPC codes over this larger class of structures. For this simulation, the non-zero entries of the LDPC matrix are chosen from the set of non-zero-divisors of $GR(8, 2)$. The constraint graph structure and the cardinality of the $GR(8, 2)$-LDPC code is the same as the $(3, 12)$ $\mathbb{G}_{64}$-LDPC code, and so is the performance with BP decoding when ring elements are mapped to $2 \times S_o$.

All the codes described above are tabulated in Table I. Also included are the threshold values for each coded modulation scheme, computed as described in Section III-C

Using the techniques of Section IV-B, better LDPC codes over $\mathbb{G}_{64}$ can be found by optimizing the degree

Ring	Code parameters		Code rate	Code blocklength	Signal	Threshold
	j	k		(inringsymbols)	set	
$\mathbb{G}_{64}$	3	12	0.75	1000	$2 \times S_o$	2.2 dB
$GR(8,2)$	3	12	0.75	1000	$2 \times S_o$	2.25 dB
$\mathbb{G}_{64}$	3	6	0.5	1500	$2 \times$ 8PSK	3.2 dB
$\mathbb{Z}_8$	3	6	0.5	3000	8 PSK	2.5 dB
$\mathbb{Z}_8$	3	12	0.75	2000	S_o	2.3 dB

TABLE I

LIST OF CODES SIMULATED WITH EFFECTIVE RATE 1.5 BITS/S/HZ AND 3000 2D-SYMBOLS.

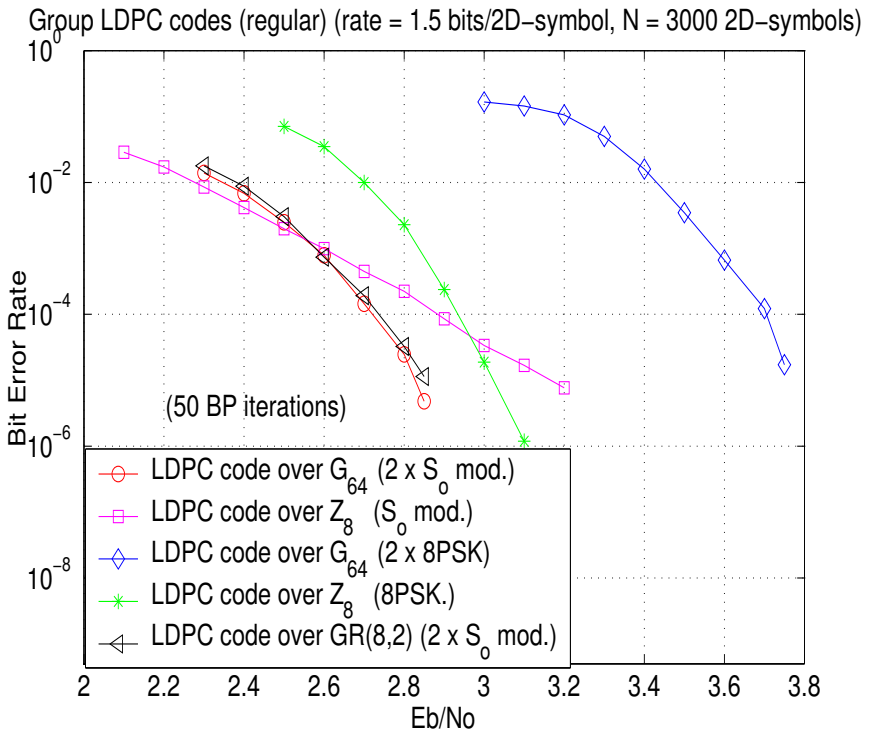

Fig. 9. Group LDPCs matched to multi-dimensional constellations. (Rate=1.5 bits/2D-symbol)

profile with respect to the resulting threshold. (Because of the complexity of BP decoding over $\mathbb{G}_{64}$, a much narrower range of degree profiles was searched; specifically, $\rho(x)$ was fixed to be "concentrated" as before, but $\lambda(x)$ was constrained such that $\deg[\lambda(x)] \leq 2$.) Among the degree profiles considered, the following was found to yield the best threshold among LDPC codes over $\mathbb{G}_{64}$ with a rate of 1.5 bits/2D-symbol when used with $2 \times S_8$ modulation:

$$\lambda(x) = 0.29091x + 0.70909x^2 \quad \text{and} \quad \rho(x) = 0.5x^9 + 0.5x^{10}, \tag{7}$$

The threshold for this degree distribution is $E_b/N_0 = 2.0$ dB. Figure 10 shows the performance of the irregular LDPC code over $\mathbb{G}_{64}$ with the degree profile as given in (7). There is an improvement (of about 0.2 dB) in the performance of the irregular code compared to the regular $(3, 12)$ code from Section V. The performance of regular $\mathbb{Z}_8$ codes from Section V are also shown for reference. All the codes have an effective block length of 3000 2D symbols and a rate of 1.5 bits/2D-symbol. Note however, that the irregular LDPC code over $\mathbb{Z}_8$ in Figure 6 at 1.5 bits/symbol performs better than the irregular $\mathbb{G}_{64}$ LDPC code in Figure 10. This is probably because the degree distribution of the $\mathbb{Z}_8$ code is much better than that of the $\mathbb{G}_{64}$ code, since the degree profile search for the $\mathbb{Z}_8$ code was much wider than that of the $\mathbb{G}_{64}$ code.

VI. Conclusions

This paper has investigated the structure and design of bandwidth-efficient coded modulation based on LDPC codes over integer rings mapped to "matched" PSK-like signal sets. Many of the issues that are prominent in the binary LDPC code literature – e.g., threshold calculation, regular/irregular code design, etc. – have analogous roles with respect to LDPC codes over these more general structures. Using LDPC codes over $\mathbb{Z}_8$ mapped to 8-PSK as a motivating example, it was shown that the proposed approach provides coding gain over coded modulation based on *binary* LDPC codes – i.e., bit-interleaved coded modulation.

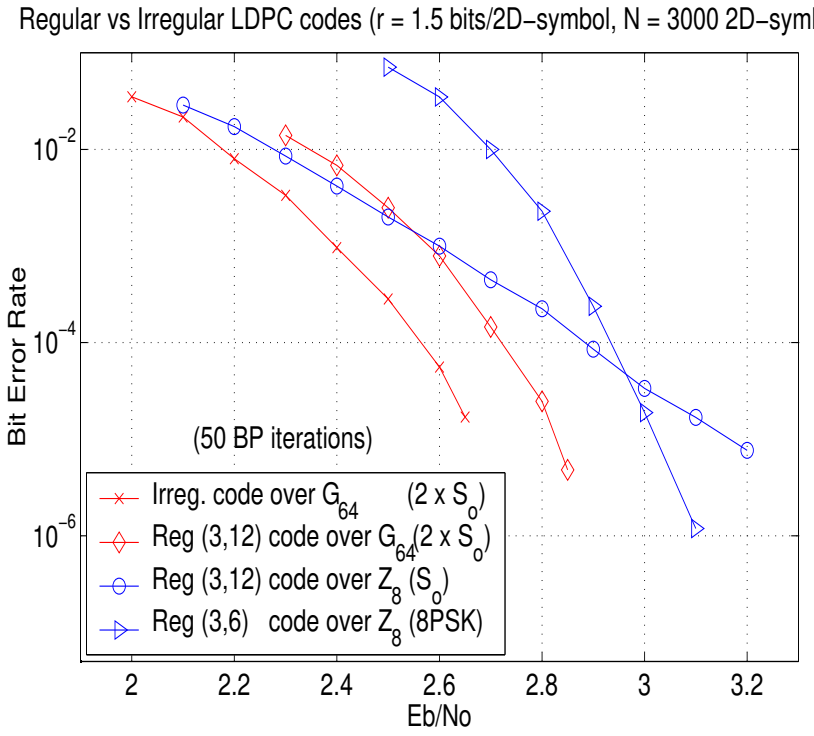

Fig. 10. Regular vs irregular group LDPC codes. (Rate=1.5 bits/2D-symbol)

APPENDIX

Proof: As per Section III-A, without loss of generality assume H is of the form

$$H = \begin{bmatrix} B_{0,a} & B_{0,a-1} & \dots & B_{0,1} & I_{k_a} \\ pB_{1,a} & pB_{1,a-1} & \dots & pI_{k_{a-1}} & 0 \\ . & . & \dots & . & . \\ p^{a-1}B_{a-1,a} & p^{a-1}I_{k_1} & \dots & 0 & 0 \end{bmatrix} \tag{8}$$

where $n = k_0 + k_1 + k_2 + \ldots + k_a$. Then the parity check matrix of C_p is given by

$$H_p = [\tilde{B}_{0,a} \ \tilde{B}_{0,a-1} \ \dots \ \tilde{B}_{0,1} \ I_{k_a}],$$

where $\tilde{B}_{0,i} \in \mathbb{Z}_p$ and $\tilde{B}_{0,i} = B_{0,i} \bmod p$, for $1 \leq i \leq a$.

Let $\mathbf{x}$ be a non-zero codeword in C_p. We will show that there is a codeword in C with the same Hamming weight as $\mathbf{x}$, which in turn implies $d_H(C) \leq d_H(C_p)$. Specifically, consider $\mathbf{y} = p^{a-1}\mathbf{x} \in \mathbb{Z}_{p^a}^n$. If x_i is a non-zero component of $\mathbf{x}$ then $0 < x_i < p$ implies that $p^{a-1}x_i$ is not a multiple of p^a and so $y_i \neq 0$; therefore, $wt_H(\mathbf{x}) = wt_H(\mathbf{y})$. Moreover,

$$H_p\mathbf{x}^T = \mathbf{0} \bmod p, \quad \text{(by hypothesis)}$$

$$\Rightarrow [\tilde{B}_{0,a}\ \tilde{B}_{0,a-1}\ \ldots\ \tilde{B}_{0,1}\ I_{k_a}]\mathbf{x}^T = \mathbf{0} \bmod p$$

$$\Rightarrow [\tilde{B}_{0,a}\ \tilde{B}_{0,a-1}\ \ldots\ \tilde{B}_{0,1}\ I_{k_a}](p^{a-1}\mathbf{x})^T = \mathbf{0} \bmod p^a$$

$$\Rightarrow [B_{0a}\ B_{0,a-1}\ \ldots\ B_{01}\ I_{k_a}](p^{a-1}\mathbf{x})^T = \mathbf{0} \bmod p^a$$

since $p^{a-1}b = p^{a-1}\tilde{b}$ in $\mathbb{Z}_{p^a}$ if b is an element of $\mathbb{Z}_{p^a}$ and $\tilde{b}$ is the element $\mathbb{Z}_p$ equivalent to b modulo p.

From this we observe that $\mathbf{y} = p^{a-1}\mathbf{x}$ satisfies the parity constraints in the first k_a rows in H; it also obviously satisfies the other parity check constraints since $p^i\mathbf{y} = 0$ in $\mathbb{Z}_{p^a}$ for $i \geq 1$. Therefore $\mathbf{y} \in C$ and so $d_H(C) \leq d_H(C_p)$.

Now assume C is free; we wish to show $d_H(C) = d_H(C_p)$. This will be done by proving two claims:

- Claim 1: There is a minimum-weight codeword in C with non-zero components that are all non-zero-divisors.
- Claim 2: any codeword in C reduced modulo p is a codeword in C_p.

It's obvious that these two claims will prove $d_H(C_p) \leq d_H(C)$; if the minimum-weight codeword whose existence is postulated in Claim 1 is reduced modulo p, the result will be a non-zero codeword of the same weight in C_p. Since we have already shown that $d_H(C) \leq d_H(C_p)$, the two claims also prove the theorem.

To prove Claim 1, assume the parity check matrix of C is of the form $H = [B\ I_{k_a}]$. (This assumption is valid since C is free). Then $G = [I_{n-k_a}\quad -B^T]$ is a generator matrix for C and any codeword in C can be

written as $\mathbf{u}G$ for some $\mathbf{u} \in (\mathbb{Z}_{p^a})^{n-k_a}$. Let $\mathbf{y}$ be a minimum weight codeword (of Hamming weight d_{min}) in C and assume at least one component of $\mathbf{y}$ is a non-zero-divisor; then all d_{min} non-zero components in $\mathbf{y}$ must be non-zero-divisors. Why? Because otherwise the codeword $p^{a-1}\mathbf{y}$ would have a Hamming weight less than d_{min}, contradicting our assumption. Therefore a minimum-weight codeword that contains *at least one* non-zero-divisor must contain *all* non-zero-divisors.

It remains to be shown that there is a minimum-weight codeword with at least one non-zero-divisor. Suppose $\mathbf{y}$ is a minimum weight codeword and suppose all the components of $\mathbf{y}$ are equivalent to zero mod p. If some of the non-zero components of $\mathbf{y}$ are different from the element p^{a-1}, then multiply $\mathbf{y}$ by the ring element p and call this the new codeword $\mathbf{y}$. If there is still a non-zero component in the new codeword that is not p^{a-1}, then once again multiply the new codeword by the ring element p. Repeat this operation until all the non-zero components of the result – call it $\mathbf{y}^*$ – are p^{a-1}. Since $\mathbf{y}^* \neq \mathbf{0}$, the Hamming weight of $\mathbf{y}^*$ is still d_{min}. Moreover, $\mathbf{y}^* = \mathbf{u}G$ where all the components of $\mathbf{u}$ are either 0 or p^{a-1}; therefore $p^{a-1}|\mathbf{u}$. Let $\mathbf{v} = \mathbf{u}/p^{a-1}$, i.e., the components of $\mathbf{u}$ that are p^{a-1} are replaced by the ring element 1. Form the codeword $\mathbf{z} = \mathbf{v}G$. Since $\mathbf{v} \neq \mathbf{0}$ and G is in systematic form, the codeword $\mathbf{z}$ has at least one non-zero-divisor. This proves Claim 1.

Claim 2 is proved by noting that if $\mathbf{c} \in C$ then $H\mathbf{c}^T = 0 \bmod p^a$ and so $H\mathbf{c}^T = 0 \bmod p$. This in turn implies that $(H \bmod p)(\mathbf{c}^T \bmod p) = \mathbf{0} \bmod p$ and so $\mathbf{c} \bmod p$ is a codeword in C_p.

REFERENCES

[1] K. Narayanan and J. Li, "Bandwidth efficient low density parity check coding using multilevel coding and iterative multistage decoding," in *Proceedings of the International Symposium on Turbo Codes and Related Topics*, (ENST-Bretagne, Brest, France), pp. 165–168, Sept. 2000.

[2] J. Hou, P. H. Siegel, B. Milstein, and H. D. Pfister, "Design of low-density parity-check codes for bandwidth efficient modulation," in *Proceedings of IEEE Information Theory Workshop*, (Cairns, Australia), pp. 24–26, October 2001.

[3] D. Slepian, "Group codes for the Gaussian channel," *The Bell System Technical Journal*, pp. 575–602, April 1968.

[4] G. D. Forney, "Geometrically uniform codes," *IEEE Transactions on Information Theory*, pp. 1241–1260, Sept. 1991.

[5] H. A. Loeliger, "Signal sets matched to groups," *IEEE Transactions on Information Theory*, pp. 1675–1682, Nov 1991.

[6] H. A. Loeliger, *On Euclidean Space Group Codes*. PhD thesis, Swiss Federal Institute of Technology, Zurich, 1992.

[7] G. Caire and E. M. Biglieri, "Linear block codes over cyclic groups," *IEEE Transactions on Information Theory*, pp. 1246–1256, Sept 1995.

[8] D. J. C. Mackay and M. Davey, "Low density parity check codes over $GF(q)$," *IEEE Communication Letters*, vol. 2, no. 6, pp. 165–167, June 1998.

[9] A. R. Calderbank and N. J. A. Sloane, "Modular and p-adic cyclic codes," *Designs, Codes, and Cryptography*, no. 6, pp. 21–35, 1995.

[10] R. G. Gallager, *Low-Density Parity-Check Codes*. Cambridge, Massachusetts: MIT Press, 1963.

[11] D. Sridhara, *Bandwidth Efficient Coded Modulation Using Low Density Parity Check Codes*. PhD thesis, Department of Electrical Engineering, University of Notre Dame, May 2003.

[12] T. Richardson and R. Urbanke, "The capacity of low-density parity check codes under message-passing decoding," *IEEE Transactions on Information Theory*, pp. 599–618, Feb 2001.

[13] S.-Y. Chung, T. Richardson, and R. Urbanke, "Analysis of sum-product decoding of low density parity-check codes using a Gaussian approximation," *IEEE Transactions on Information Theory*, pp. 657–670, Feb 2001.

[14] S. ten Brink, "Designing iterative decoding schemes with the extrinsic information transfer chart," *AEU International Journal of Electronics and Communications*, vol. 54, no. 6, pp. 389–398, Nov 2000.

[15] S.-Y. Chung, G. D. Forney, T. J. Richardson, and R. Urbanke, "On the design of low-density parity-check codes within 0.0045 dB of the Shannon limit," *IEEE Communication Letters*, no. 5, pp. 58–60, 2001.

[16] G. Bini and F. Flamini, *Finite Commutative Rings and Their Applications*. Boston, Massachusetts: Kluwer Academic Publishers, 2002.

[17] D. Sridhara and T. E. Fuja, "Bandwidth efficient modulation based on algebraic low density parity check codes," in *Proceedings of IEEE International Symposium on Information Theory*, (Washington, D.C.), p. 165, June 2001.

Biography

Deepak Sridhara received the B. Tech degree in electrical engineering from the Indian Institute of Technology, Madras, in 1998, the M.S. and Ph.D degrees in electrical engineering from the University of Notre Dame, IN, in 2000 and 2003, respectively.

Between January and December 2004, he worked as a post-doctoral research associate in the Department of Mathematics at the Indian Institute of Science, Bangalore, India. He is currently a post-doctorate at the Institute for Mathematics at the University of Zurich. His research interests include coding theory, codes over graphs and iterative techniques, and information theory.

Tom Fuja received his undergraduate education at the University of Michigan, graduating with a B.S.E.E. and a B.S.Comp.E. in 1981. He subsequently attended Cornell University, where he received the M.Eng. and the Ph.D. degrees in electrical engineering in 1983 and 1987, respectively.

In 1998 Prof. Fuja joined the faculty of the University of Notre Dame in South Bend, IN, where he is a professor of electrical engineering; from 1987 to 1998, Fuja was on the faculty of the University of Maryland in College Park, MD. In addition, Prof. Fuja served as Program Director for Communications Research at the U.S. National Science Foundation in 1997 and 1998. He was the General Co-Chair of the 2001 International Symposium on Information Theory in Washington DC, and in 2002 he served as the President of the IEEE Information Theory Society.

Prof. Fuja's research interests lie in coding theory and applications and information theory. Most of his recent research has focused on coding for wireless applications and on the interface between source coding and channel coding.